ZABBIX '26
CONFERENCE
POLAND

Saving Private Ryan

"The mission is a man"

ZABBIX '26
CONFERENCE
POLAND

Saving the Master

Automated PostgreSQL recovery with Zabbix and Event-Driven Ansible

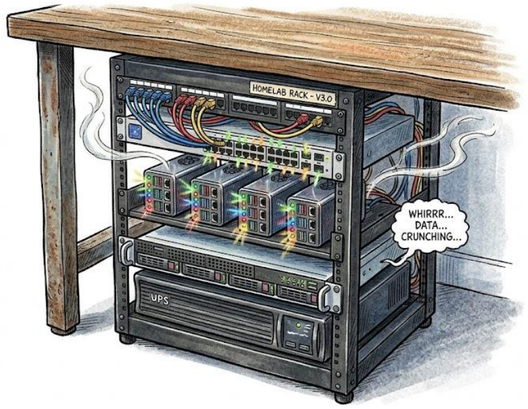
Maksym Buz

Senior Support Engineer @ Zabbix

Certified Expert and Trainer

Homelab enthusiast

Bicycle maniac



ZABBIX '26 CONFERENCE POLAND

Chapter 1: The Problem

ZABBIX '26
CONFERENCE
POLAND

ZABBIX

The Power Trio

EDA

Event-handling capability needed to automate time-consuming tasks and respond to changing conditions in any IT domain.

Zabbix

An enterprise-class open open source distributed distributed monitoring monitoring solution.

"Your business
You know it."

PostgreSQL + Patroni

A powerful database for the most complicated workloads, managed by the Patroni High Availability template.

The WAL Bloat Nightmare

Replica:

Experiences high latency.

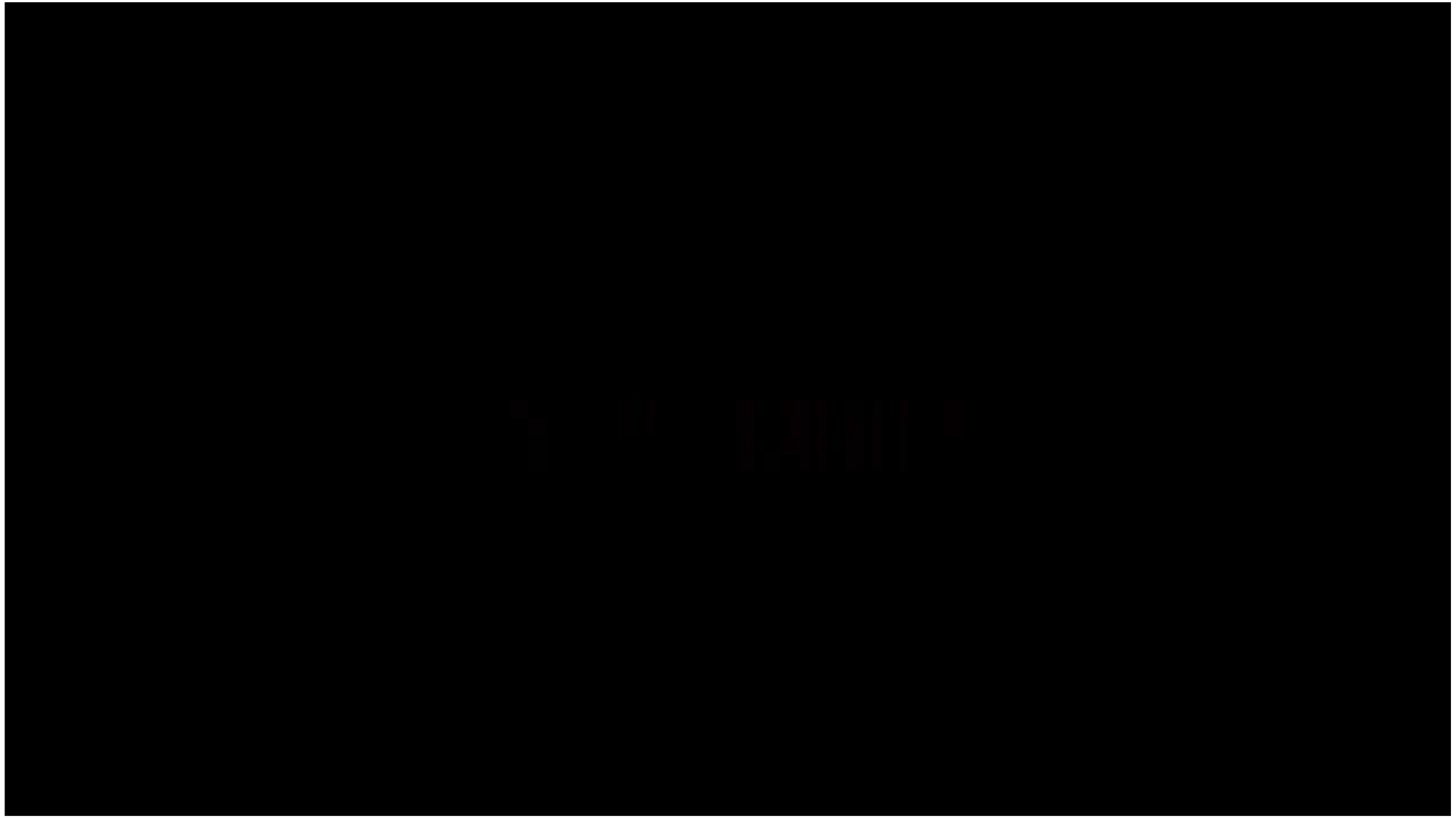
Patroni:

This is fine. Process is running.

Master:

I must keep all WAL files for my slow brother!

A few hours later:



Why HA is a lie (sometimes)

max_slot_wal_keep_size ?

Yes

- The master survives

But

- Replication is permanently broken
- HA Cluster is degraded
- Manual intervention is required to rebuild replica

Chapter 2: The Solution

ZABBIX '26
CONFERENCE
POLAND

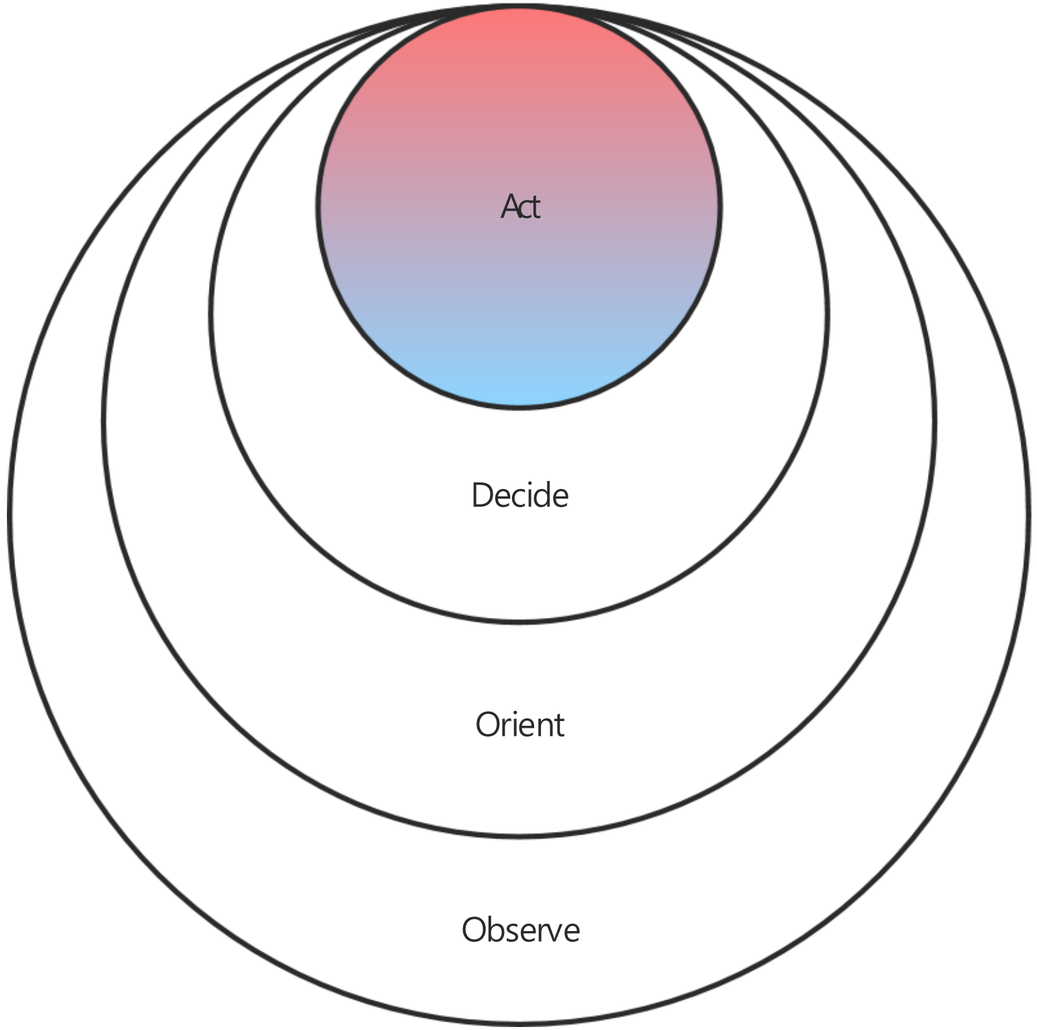
Self-healing structure

Observe (Zabbix): Real-time monitoring and event detection

Orient (EDA): Event processing and context enrichment via Event-Driven Ansible

Decide (Rulebook): Selection of the remediation strategy based on predefined logic

Act (Playbook): Automated execution of recovery tasks to restore the desired state



The Philosophy of Zero-Touch Recovery

Proactive Sensing

Correlates WAL growth with space available. Prevents 100% disk usage

The Quick Swap

Drops replication slot, destroys container. Instantly saves Primary

Restoring Redundancy

Automated spin-up of fresh PostgreSQL container. Restores full HA

Chapter 3: The Process

ZABBIX '26
CONFERENCE
POLAND

Smart Detection Logic

Trigger expression:

last(/PostgreSQL/pgsql.replication.lag.b) > 10G and last(/PostgreSQL/vfs.fs.size[/var/lib/postgresql,pfree]) < 10

Capacity aware

Fires only when lag threatens disk

Predictive

Catches high-velocity WAL growth.

Reliable

Ignore slow replica that doesn't impact the Master.

The Precision Webhook: Tags & Context

```
{
  "trigger_name": "Critical WAL buildup on pg-node-02",
  "trigger_status": "PROBLEM",
  "tags": {
    "db_node_name": "pg-node-02",
    "cluster_name": "zbx-cluster"
  },
  "event_id": "100532"
}
```

Power of Tags

Using tags for surgical

Flexible Payload

Total control of the dataflow.

Actionable

Transforming a static alert into automation.

EDA Logic: Intelligence in Action

```
  # 1. Rulebook: Capture the Zombie
  action:
    run_playbook:
      name: rip_and_tear.yml
      extra_vars:
        db_node: "{{ event.payload.tags.db_node_name }}"

  # 2. Playbook: Locate the Leader
  set_fact:
    db_master: "{{ members | selectattr('role', 'equalto', 'leader') |
map(attribute='name') | first }}"
```

Zero Guesswork

Zabbix tags directly inject the exact failing node name into the automation pipeline.

Dynamic Discovery

Ansible queries the Patroni API to current active Master.

Bulletproof Logic

Separating the alert from state prevents deletion of the wrong node.

Phase 1: Triage & Amputation

```
# 1. Stop the Bleeding
   - name: Drop replication slot on Primary
     docker_container_exec:
       container: "{{ db_master }}"
       command:
        psql -U postgres -c
        "SELECT pg_drop_replication_slot('{{ db_node_name }}');"
```

```
# 2. Kill the Zombie
  - name: Remove broken replica container
    docker_container:
      name: "{{ db_node_name }}"
      state: absent
```

The Master is Safe

Dropping the slot instantly clears WAL bloat bloat

Zero Tolerance

We don't troubleshoot the broken node, we terminate it

Phase 2: Scorched Earth & Respawn

```
# 3. Scorched Earth
 - name: Delete volume to force fresh basebackup
  docker_volume:
    name: "pgdata_{{ db_node_name }}"
    state: absent
```

```
# 4. The Respawn
 - name: Recreate and join replica to cluster
  command: docker compose up -d {{ db_node_name }}
```

Clean Slate

Destroying the volume guarantees no data

Auto-Healing

Patroni automatically triggers a fresh

The Result? A Symphony of Logs

```
[ZABBIX] PROBLEM: Critical WAL buildup on pg-node-02 (Lag: 12GB)
[ZABBIX] ACTION: Sending Webhook to EDA payload... [OK]
[EDA] EVENT: Payload matched rule 'Respond to WAL Crisis'
[EDA] TASK: psql -c "SELECT pg_drop_replication_slot('pg-node-02')" -> CHANGED
[EDA] TASK: docker rm -f pg-node-02 -> CHANGED
[EDA] TASK: docker volume rm pgdata_pg-node-02 -> CHANGED
[EDA] TASK: docker-compose up -d pg-node-02 -> CHANGED
[PATRONI] INFO: starting node pg-node-02...
[PATRONI] INFO: running pg_basebackup from pg-node-01 (Leader)
[PATRONI] INFO: pg_basebackup completed successfully.
[PATRONI] INFO: Lock owner: pg-node-01; I am pg-node-02 (role: replica)
[ZABBIX] RESOLVED: Critical WAL buildup on pg-node-02
```

The Limitless Horizon

Beyond Postgres

This is just the starting line.

Limitless Scale

Networks, Clouds, Apps — automate the universe.

The Golden Rule

If it can be monitored, it can be healed.

ZABBIX

Keep pedaling...

Thank you!

Questions?